***** DRAFT: 12/20/98 *****

SNS INTEGRATED CONTROL SYSTEM
SOFTWARE QUALITY ASSURANCE PLAN

1. PURPOSE

This plan describes how quality assurance will be implemented for the SNS Integrated Control System (ICS) software.

2. SCOPE

This plan addresses what needs to be done to ensure development of highly-reliable ICS software required for efficient operation and maintenance of the SNS facility. Detailed descriptive information will be provided in other documents, as needed, including a Software Development Plan, development environment and tools, roles and responsibilities of development staff, security, databases. This plan identifies a minimum set of documents and describes the content of each.

Most of the software that will be used for SNS control already exists as the EPICS software distribution. New software will be written to augment and extend this software. Consequently the QA and configuration management plans and work will reflect a heavy emphasis on the operation, maintenance, and testing phases of the traditional software development life cycle.

Since the ICS will be prohibited from performing safety class or safety significant functions, software written for this purpose will be covered by a different QA plan. ICS software will perform a "production support" function.

This plan will cover elements common to all categories of software used in the ICS. A number of software QA elements will vary according to the type of software. Individual QA plans for each category will address elements which depend on the requirements unique to each.

3. REFERENCE DOCUMENTS

U.S. DOE 200.1. *Software Engineering Methodology (year).*

LANL APT Software Development Plan

[Reference to a good reference text, e.g., Roger Pressman book.]

[List family of IEEE software development standards that apply. This document follows most closely the IEEE standard for software QA plans.]

4. MANAGEMENT *[Of the Execution of This Plan]*

4.1. Organization

[Depict organizational structure that influences quality of the software, i.e., who is responsible for SQA and dependence/independence from development process.]

4.2. Tasks

[Describe tasks associated with implementation of this plan and sequence in which they must be done.]

4.3. Roles and Responsibilities

4.3.1. SNS Quality Assurance Manager – Approves this plan and has final responsibility for its use. *[Copied from APT plan; is it what we want?]*

4.3.2. Senior Team Leader (STL) – Oversees the ICS software QA program and ensures that the third level task leaders are cognizant of program requirements.

4.3.3. **ICS QA Representative** – periodically audits the ICS software QA program and strives for continuous QA program improvement. Findings are reported to the STL.

4.3.4. **Third-Level Task Leaders** – Responsible for preparing individual QA plans and obtaining the required approvals.

4.3.5. **Configuration Manager** – Establishes and maintains software configuration management systems.

4.3.6. **Developers** – Prepare the products of the lifecycle phases and participate in the quality assessment of these products.

4.3.7. **Customers** – Provide requirements and participate in product reviews.

5. DOCUMENTATION

5.1. Purpose

Success on a project the size of SNS depends critically on timely, clear communication of information. This will be especially important with contributions coming from several different laboratories, each with its own culture and approach to software development and QA. It will be very important therefore to identify a common set of requirements that all parties are prepared to support. Somewhat more documentation will be required, as well as possible departures from the standard approaches recommended for software development projects.

A proposed flow of documentation for the SNS control system software is illustrated in Fig. 5.1.

[Figure 5.1 will illustrate the document flow indicated in the outline below. It is intended to clarify how the various documents are related (show dependencies) to one another.]

+++ Start Figure 5.1 +++

[These documents provide the high level guidance, so need to be prepared very early in the project.]

SNS Software QA Plan for Control System Software (General)

SNS Control System Software Development Plan

SNS Control System Software Configuration Management Plan

SNS Control System Standards and Procedures Manual

[Content of the following documents fills in detail not covered in the general plan and needed for that category of software, primarily in the areas of documentation, standards and practices, reviews and audits, problem reporting and corrective action. Details would be provided as the need arises (if no new EPICS extensions are developed, a QA Plan would not be prepared for this category of software).]

EPICS System Software QA Plan (Category 1)

EPICS Extensions Software QA Plan (Category 2)

EPICS Applications Software QA Plan (Category 3)

[While a single QA plan would be written to deal with Category 3 software development, each application (including databases) will require its own set of documents covering topics such as

requirements

design specifications

software verification and validation plan

implementation description

testing procedures

with levels of detail and amount of testing commensurate with the significance of the application to the operation of SNS.]

+++ End Fig. 5.1 +++

[Contents of this section (5) identify documentation governing the development, verification and validation, use, and maintenance of the software. Include statements about how the documents are to be checked for adequacy and identification of the review or audit by which adequacy of each document shall be confirmed.]

5.2. General Development Environment

5.2.1. Software Development Plan

[Discuss lifecycle phases in this plan (see Appendix A for comments/questions about life cycle phase considerations that need to be addressed). Describe the Open System Software development model (examples are GNU, Linux OS, developed within a loosely knit, but dedicated technical community), not generally recognized by the software development experts, but a force to be reckoned with. Describe the EPICS collaboration and how it functions (its role in the development life cycle), use of tech-talk and associated archives. Describe role of ORNL software development laboratory.]

5.2.2. Software Configuration Management Plan
5.2.3. Standards and Procedures Manual

5.3. Software Development by Category

[These sections will define each category and QA requirements that must be met by each and which will govern the need for documentation (how much, what kind, etc.). QA plans will be developed for each of these categories to provide more detailed and specific guidance needed for that category, i.e., supplements to general guidance given in this plan.]

5.3.1. Category 1: EPICS System Software (Base)
5.3.2. Category 2: EPICS Extensions
5.3.3. Category 3: EPICS Applications

[Develop QA plan for this category since much of the development work will probably fall in this area. This category may have to be subdivided along "major subsystem" lines. Need to discuss this with ICWG. Would expect applications to be identified by IOC.]

5.3.4. Category 4: Commercial Software
5.3.5. Category 5: Public Domain and Other Third Party Software

5.4. Minimum Documentation Required

5.4.1. Category 1: EPICS System Software (Base)
[List documents; describe contents]

5.4.2. Category 2: EPICS Extensions
[List documents; describe contents]

5.4.3. Category 3: EPICS Applications
[List documents; describe contents]

5.4.4. Category 4: Commercial Software
[List documents; describe contents]

5.4.5. Category 5: Public Domain and Other Third Party Software
[List documents; describe contents]

6. STANDARDS, PRACTICES, AND CONVENTIONS
    6.1. Purpose
    6.2. Required Common Elements
    6.3. Documentation
    6.4. Logic Structure
    6.5. Coding
    6.6. Commentary

7. REVIEWS AND AUDITS
    7.1. Purpose
    7.2. Minimum Requirements

[Pick items from the list below as appropriate for QA plan for category (e.g., categories 1-5) of software considered. More items are needed for development of EPICS Base software than many EPICS applications, depending on consequences of errors.]

        7.2.1. Software Requirements Review
        7.2.2. Preliminary Design Review
        7.2.3. Critical Design Review
        7.2.4. Software Verification and Validation Review
        7.2.5. Functional Audit
        7.2.6. Physical Audit
        7.2.7. In-Process Audits
        7.2.8. Managerial Reviews

8. SOFTWARE CONFIGURATION MANAGEMENT
    8.1. CVS Server
    8.2. EPICS System Software (Base and Extensions)
    8.3. EPICS Applications
    8.4. Commercial Software
    8.5. Other Third Party Software

[Defer details to a Software Configuration Management Plan. Keep discussions here at a high level.]

9. PROBLEM REPORTING AND CORRECTIVE ACTION
[Include information about specific organizational responsibilities related to implementation of procedures and practices in this area.]

10. TOOLS, TECHNIQUES, AND METHODS
[What items will be used to support QA? State their purpose; describe their use.]

11. CODE CONTROL
    11.1. CVS Server Operation

12. MEDIA CONTROL
    12.1. CVS Server Security
    12.2. Development Workstation Security

13. SUPPLIER CONTROL
    13.1. Commercial Software
    13.2. Public Domain and Other Third Party Software

14. RECORDS COLLECTION, MAINTENANCE, AND RETENTION

[Describe what documentation from the development efforts must be retained for long periods and how this is to be done. Prefer as much documentation as possible be maintained in electronic form and Internet accessible. Need to archive manuals for all procured software as part of SNS records.]

APPENDIX A: SOFTWARE DEVELOPMENT LIFE CYCLE PHASE ISSUES TO ADDRESS

[The life cycle phase activities described below should be tailored to the development needs for each category of software. These needs (especially with regard to documentation) will depend on many factors, such as

How critical to the operation of SNS the proposed software will be
How many different people will have to use and maintain the software
Level of training of people who must use and maintain the software
How long the software is expected to be in use
How the proposed software may affect reliable operation of other software]

[Once required documents and activities (including reviews and approvals) for each phase have been identified for each category of software, prepare summary tables to serve as a checklist to indicate the status of each development effort. If desired, these checklists could be posted so they are Internet accessible to other project members. These tables will identify the products (deliverables) for each phase, who does the work, who reviews the products, and who approves them.]

A.1 CONCEPTUAL DESIGN

[This phase covers prototyping and demonstration. Documentation should address the following topics:

Major objectives
High level requirements
Who will do the work
Training plans
Discussion/review by EPICS collaboration, ICWG, etc.
Cost and schedule estimates
Prototyping/demonstration activities

Need to define an approval process that will determine whether to proceed to the next phase. In this and all other phases a mechanism needs to be set up for reporting and resolving problems.]

A.2 REQUIREMENTS SPECIFICATION

[Prepare Software Requirements Specification. Identify testing and performance requirements that must be satisfied. Define approval process to determine whether to proceed to the next phase.]

A.3 FUNCTIONAL DESIGN

[Prepare Functional Design Description which must include topics such as

User interface including preliminary CRT displays
Interfaces to the rest of the system
I/O list

Develop a back-up and recovery plan. Identify further testing that is required to meet design requirements. Need to revise cost and schedule estimates. Define approval process to determine whether to proceed to the next phase.]

A.3 SYSTEM DESIGN

[Prepare System Design Description, Software Verification and Validation Plan, Software Configuration Management Plan (or addendum to a general plan), and Test Plan. Identify any hardware required to implement software.]

A.4 PROGRAMMING

[Implement Software Configuration Management Plan. Develop documentation/annotations to describe algorithms used. Conduct unit tests. Begin preparation of Programmer's and User's Reference Manuals.]

A.5 INTEGRATION AND TESTING

[Prepare Software Verification and Validation Report summarizing results of tests. Prepare training materials, as required.]

A.6 INSTALLATION AND ACCEPTANCE

[Set up a problem reporting and resolution procedure for the customer to use. Conduct acceptance tests and document results.]

A.7 MAINTENANCE AND OPERATION

[Implement problem reporting and resolution procedure.]

A.8 DECOMMISSIONING

Contributors:

Bill Devan
John Munro